Clasificación Binaria

Estudiantes de Portugués

Sergio Del Castillo Baranda

4/10/2020

Carga de los datos y librerías

Cómo dataset se ha escogido un conjunto de alumnos de un colegio, los datos muestran información sobre cada uno de los alumnos en lo relativo a su vida personal (dirección de vivienda, trabajo de sus padres, tamaño de la familia. . .) y datos relativos a sus interacciones con el estudio (Tiempo de estudio, actividades extracurriculares, tiempo que dedica el alumno al estudio. . .).

El objetivo de esta práctica será averiguar el conjunto de alumnos que van a realizar estudios superiores con la información que tenemos en el dataset.

A continuación mostraré un poco de información sobre los datos contenidos en el documento:

```r
students.csv <- file.path(getwd(), 'student-por.csv')
STUDENTS <- read.csv2(file = students.csv, header = TRUE, sep = ';')

summary(STUDENTS)
```

```
##  school   sex          age        address famsize   Pstatus
##  GP:423   F:292   Min.   :15.00   R:127   GT3:358   A: 61
##  MS: 76   M:207   1st Qu.:16.00   U:372   LE3:141   T:438
##                   Median :16.00
##                   Mean   :16.58
##                   3rd Qu.:17.00
##                   Max.   :22.00
##       Medu            Fedu             Mjob           Fjob
##  Min.   :0.000   Min.   :0.000   at_home : 93   at_home : 23
##  1st Qu.:2.000   1st Qu.:1.000   health  : 41   health  : 19
##  Median :3.000   Median :2.000   other   :198   other   :293
##  Mean   :2.591   Mean   :2.385   services:108   services:132
##  3rd Qu.:4.000   3rd Qu.:3.000   teacher : 59   teacher : 32
##  Max.   :4.000   Max.   :4.000
##         reason      guardian     traveltime      studytime
##  course    :209   father:117   Min.   :1.000   Min.   :1.000
##  home      :128   mother:351   1st Qu.:1.000   1st Qu.:1.000
##  other     : 38   other : 31   Median :1.000   Median :2.000
##  reputation:124                Mean   :1.493   Mean   :1.976
##                                3rd Qu.:2.000   3rd Qu.:2.000
##                                Max.   :4.000   Max.   :4.000
##     failures      schoolsup famsup     paid     activities nursery
##  Min.   :0.0000   no :438   no :178   no :470   no :246    no : 99
##  1st Qu.:0.0000   yes: 61   yes:321   yes: 29   yes:253    yes:400
##  Median :0.0000
```

```
##  Mean   :0.1864
##  3rd Qu.:0.0000
##  Max.   :3.0000
##  higher    internet  romantic      famrel        freetime
##  no : 49   no :103   no :327   Min.   :1.00   Min.   :1.000
##  yes:450   yes:396   yes:172   1st Qu.:4.00   1st Qu.:3.000
##                                Median :4.00   Median :3.000
##                                Mean   :3.94   Mean   :3.198
##                                3rd Qu.:5.00   3rd Qu.:4.000
##                                Max.   :5.00   Max.   :5.000
##      goout            Dalc            Walc           health
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:2.000
##  Median :3.000   Median :1.000   Median :2.000   Median :4.000
##  Mean   :3.158   Mean   :1.483   Mean   :2.251   Mean   :3.551
##  3rd Qu.:4.000   3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:5.000
##  Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :5.000
##     absences            G1              G2              G3
##  Min.   : 0.000   Min.   : 0.00   Min.   : 0.00   Min.   : 0.00
##  1st Qu.: 0.000   1st Qu.:10.00   1st Qu.:10.00   1st Qu.:11.00
##  Median : 2.000   Median :12.00   Median :12.00   Median :12.00
##  Mean   : 3.948   Mean   :11.74   Mean   :11.89   Mean   :12.33
##  3rd Qu.: 6.000   3rd Qu.:13.50   3rd Qu.:13.00   3rd Qu.:14.00
##  Max.   :32.000   Max.   :18.00   Max.   :19.00   Max.   :19.00
```

SELECCIÓN DE VARIABLES

El objetivo de este apartado es obtener las mejores variables que nos permitan optimizar nuestro modelos. El trabajo lo realizaremos en dos fases, una fase inicial en la que vamos a realizar una limpieza de datos para obtener un dataset con el que podamos generar un modelo y en segundo lugar lo que realizaremos selección de las mejores variables para optimizar nuestro modelo.

LIMPIEZA DE NA

No realizamos supresión de NA dado que no hay ninguno en el fichero.

```r
check.na(STUDENTS)
```

```
##
## There is a total of  0  NAs on this file
```

```
## [1] 0
```

Para comenzar a trabajar con las variables vamos a hacer una selección en función del tipo de variable que es, a continuación trabajaremos con las variables de forma diferente en función de la clase de variable que sea.

Lo primero que haremos será la selección de la variable objetivo (higher) y la separamos del dataset. A continuación, haremos una subdivisión de las columnas restantes entre continuas y categóricas almacenando los nombres de las columnas en dos variables.

```r
vardep <- "higher"
students.bis <- STUDENTS[,-which(names(STUDENTS) == vardep)]

continuas <- names(select_if(students.bis, is.integer))
categoricas <- names(select_if(students.bis, is.factor))

cat("Nuestra variable objetivo será: ",vardep, "\n\nVariables continuas: ",continuas, "\n\nVariables ca
```

```
## Nuestra variable objetivo será:  higher
##
## Variables continuas:  age Medu Fedu traveltime studytime failures famrel freetime goout Dalc Walc hea
##
## Variables categoricas:  school sex address famsize Pstatus Mjob Fjob reason guardian schoolsup famsu
```

CREACIÓN DE VARIABLES DUMMY

Generamos variables dummy a partir de nuestras variables categóricas. En nuestro caso generaremos variables dummies con todas dado que las variables categóricas no contienen un número demasiado elevado de valores diferentes.

```
students.df<- dummy.data.frame(STUDENTS, categoricas, sep = ".")
```

ESTANDARIZACIÓN DE VARIABLES

A continuación estandarizamos las variables continuas. Primero realizamos la media y desviación típica de las contínuas y a continuación las estandarizamos. Para trabajar ahora con todas las variables como continuas, las uno a las variables dummy generadas en el paso anterior.

```
means <- apply(students.df[,continuas],2,mean)
sds <- sapply(students.df[,continuas],sd)


students.df.bis <- scale(students.df[,continuas], center = means, scale = sds)
numerocont <- which(colnames(students.df) %in% continuas)
students.df.s <- cbind(students.df.bis, students.df[,-numerocont])
```

SELECCIÓN DE VARIABLES

El primer paso en la selección de las variables es suprimir de las variables dummy una variable, dado que esta puede ser obtenida con un cálculo del resto de las variables.

```
## Variables continuas:  age Medu Fedu traveltime studytime failures famrel freetime goout Dalc Walc hea
##
## Variables categoricas:
```

SELECCIÓN DE VARIABLES EN CLASIFICACIÓN BINARIA LOGÍSTICA

Para la selección de variables hacemos la búsqueda mediante el uso de la medida de ajuste AIC. Para ejecutar los algoritmos lo realizaremos mediante el método stepwise que que va incluyendo y sacando variables con el objetivo de optimizar la selección.

Lo que conseguimos al realizar estas operaciones es obtener el conjunto de variables que tienen mejores cualidades para predecir.

```
full<-glm(factor(higher)~., data=students.df.s, family = binomial(link="logit"))
null<-glm(factor(higher)~1, data=students.df.s, family = binomial(link="logit"))

seleccion<-stepAIC(null,scope=list(upper=full),direction="both", trace=0)

variables <- names(seleccion$coefficients)[-1]
cat("La mejor selección de variables viene dada por: ", variables)
```

```
## La mejor selección de variables viene dada por:  G1 age studytime G3 school.GP famsup.yes Mjob.healt
```

GENERACIÓN DE LOS SETS DE DATOS (train, test / Validación cruzada)

En el anterior apartado hemos obtenido las mejores variables para poder generar nuestros modelos. En este apartado lo que vamos a realizar es una división de los datos en dos sets, uno para la parte de test y otro para la parte de entrenamiento del modelo. El objetivo es utilizar el set de entrenamiento para entrenar nuestro modelo y prepararlo para la predicción y realizar pruebas para comprobar la eficacia con la que es capaz de predecir sobre nuestro set de test.

La validación de los datos la realizaremos mediante validación cruzada que lo que realiza es la selección del mejor conjunto de datos que formarán parte de cada set mediante la comprobación redundante de diferentes escenarios de manera que los datos que queden en un set y otro estén lo más balanceados posible.

Utilizaremos validación cruzada repetida dado que únicamente tenemos un set de 500 filas de datos. La generación de los sets de train y test se realiza 4 veces

```
set.seed(1234)
control<-trainControl(method = "repeatedcv",number=4,savePredictions = "all")
```

COMPARACIÓN DE MODELOS

REGRESIÓN LINEAL

Modelo con regresión lineal, este no tendrá rejilla porque no tiene hiperparámetros.

```
reg<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+famre
                data=students.df.s,
                method="glm",
                trControl=control,
                trace=FALSE)
reg
```

```
## Generalized Linear Model
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
## Resampling results:
##
##   Accuracy   Kappa
##   0.9097716  0.3866103
```

RED NEURONAL

Ahora vamos a generar un modelo con redes neuronales. Para comprobar su eficacia realizaremos diferentes tuneos hasta obtener el mejor resultado. La forma que tenemos de realizar el tuneado mediante el uso de una rejilla.

```
nnetgrid <-  expand.grid(size=c(1,2,3,5,10),
                         decay=c(0.01,0.1,0.001),
                         bag=FALSE)

rednnet<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+
                data=students.df.s,
                method="avNNet",linout = FALSE,
```

```
                 maxit=100,
                 trControl=control,
                 tuneGrid=nnetgrid,
                 repeats=5,
                 verbose=FALSE,
                 trace=FALSE)
rednnet
```

```
## Model Averaged Neural Network
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 374, 374, 374, 375
## Resampling results across tuning parameters:
##
##   size  decay  Accuracy   Kappa
##    1    0.001  0.8957581  0.3806485
##    1    0.010  0.9017419  0.3902573
##    1    0.100  0.9017742  0.3336351
##    2    0.001  0.9097903  0.4177093
##    2    0.010  0.8977419  0.3612457
##    2    0.100  0.9138065  0.4318188
##    3    0.001  0.9117903  0.4324913
##    3    0.010  0.9138065  0.4289713
##    3    0.100  0.9138226  0.4295853
##    5    0.001  0.9057903  0.4097755
##    5    0.010  0.9017903  0.3905407
##    5    0.100  0.9058065  0.3998800
##   10    0.001  0.9037903  0.4311235
##   10    0.010  0.9017903  0.3923020
##   10    0.100  0.9037742  0.3743618
##
## Tuning parameter 'bag' was held constant at a value of FALSE
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were size = 3, decay = 0.1 and bag
##  = FALSE.
```

Con todos los modelos se va a generar una función que permita obtener el mejor tuneo. Esta función permitirá obtener los hiperparámetros y la precisión obtenida con el modelo. Nos ayudará a ajustar mejor cada uno de los modelos en la parte del ensamblado de modelos.

```
bestTuneNnet <- function(nnetmodel, size=FALSE, decay=FALSE){
  # Función que ayuda a obtener el mejor resultado obtenido en un modelo NEURAL NET
  bestSize <- rednnet$bestTune$size
  bestDecay <- rednnet$bestTune$decay
  # Cojo los parámetros de la función si están establecidos
  if (size != FALSE) {bestSize <- size}
  if (decay != FALSE) {bestDecay <- decay}

  nnetmodel$results$method = nnetmodel$method
  nnetmodel$results[nnetmodel$results$size == bestSize &
```

```
                      nnetmodel$results$decay == bestDecay,]

}
```

BAGGING

```
set.seed(1234)
baggrid<-expand.grid(mtry=c(11)) # El número de variables independientes

bag<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+famre
           data=students.df.s,
           method="rf",
           trControl=control,
           tuneGrid=baggrid,
           linout=FALSE,
           nodesize=10,
           ntree=5000,
           sampsize=200,
           replace=TRUE,
           trace=FALSE)

bag$results$method = 'bagging'
bag
```

```
## Random Forest
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
## Resampling results:
##
##   Accuracy   Kappa      method
##   0.9017069  0.2866667  bagging
##
## Tuning parameter 'mtry' was held constant at a value of 11
```

RANDOM FOREST

```
set.seed(1234)
rfgrid<-expand.grid(mtry=seq(3, 11, by = 2))

rf<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+famrel
           data=students.df.s,
           method="rf",
           trControl=control,
           tuneGrid=rfgrid,
           linout = FALSE,ntree=5000,nodesize=10,
           replace=TRUE,
           importance=TRUE,
           trace=FALSE)
```

```r
rf
```

```
## Random Forest
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    3    0.9117394  0.3259238
##    5    0.9057550  0.3075506
##    7    0.9097711  0.3531916
##    9    0.9038029  0.3319069
##   11    0.9058190  0.3555749
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 3.
```

GRADIENT BOOSTING

```r
set.seed(1234)
gbmgrid<-expand.grid(n.trees=c(500,1000,2000),
                     interaction.depth=c(1,2,3),
                     shrinkage=c(0.01,0.05,0.1),
                     n.minobsinnode=c(20,30,40))

gbm<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+famr
            data=students.df.s,
            method="gbm",
            trControl=control,
            tuneGrid=gbmgrid,
            distribution="bernoulli",
            bag.fraction=1,
            verbose=FALSE)

gbm
```

```
## Stochastic Gradient Boosting
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
## Resampling results across tuning parameters:
##
##   shrinkage  interaction.depth  n.minobsinnode  n.trees  Accuracy
```

```
##   0.01       1                 20              500    0.9117716
##   0.01       1                 20             1000    0.9077552
##   0.01       1                 20             2000    0.9057714
##   0.01       1                 30              500    0.9117716
##   0.01       1                 30             1000    0.9097394
##   0.01       1                 30             2000    0.9098036
##   0.01       1                 40              500    0.9137878
##   0.01       1                 40             1000    0.9097716
##   0.01       1                 40             2000    0.9077875
##   0.01       2                 20              500    0.9077391
##   0.01       2                 20             1000    0.9097552
##   0.01       2                 20             2000    0.9077711
##   0.01       2                 30              500    0.9117716
##   0.01       2                 30             1000    0.9138198
##   0.01       2                 30             2000    0.9118195
##   0.01       2                 40              500    0.9077552
##   0.01       2                 40             1000    0.9077875
##   0.01       2                 40             2000    0.9118198
##   0.01       3                 20              500    0.9097552
##   0.01       3                 20             1000    0.9077870
##   0.01       3                 20             2000    0.9098190
##   0.01       3                 30              500    0.9077552
##   0.01       3                 30             1000    0.9138195
##   0.01       3                 30             2000    0.9138676
##   0.01       3                 40              500    0.9097714
##   0.01       3                 40             1000    0.9157558
##   0.01       3                 40             2000    0.9157878
##   0.05       1                 20              500    0.9057714
##   0.05       1                 20             1000    0.9057552
##   0.05       1                 20             2000    0.9037870
##   0.05       1                 30              500    0.9118198
##   0.05       1                 30             1000    0.9097875
##   0.05       1                 30             2000    0.9138036
##   0.05       1                 40              500    0.9057714
##   0.05       1                 40             1000    0.9057875
##   0.05       1                 40             2000    0.9057875
##   0.05       2                 20              500    0.9138193
##   0.05       2                 20             1000    0.9178036
##   0.05       2                 20             2000    0.9158036
##   0.05       2                 30              500    0.9138356
##   0.05       2                 30             1000    0.9178515
##   0.05       2                 30             2000    0.9098193
##   0.05       2                 40              500    0.9138359
##   0.05       2                 40             1000    0.9138195
##   0.05       2                 40             2000    0.9137394
##   0.05       3                 20              500    0.9057867
##   0.05       3                 20             1000    0.8997545
##   0.05       3                 20             2000    0.8957545
##   0.05       3                 30              500    0.9098354
##   0.05       3                 30             1000    0.8998507
##   0.05       3                 30             2000    0.8938026
##   0.05       3                 40              500    0.9117875
##   0.05       3                 40             1000    0.9038029
##   0.05       3                 40             2000    0.9017550
```

```
##   0.10      1                   20           500     0.9057552
##   0.10      1                   20          1000     0.9037870
##   0.10      1                   20          2000     0.9077711
##   0.10      1                   30           500     0.9118036
##   0.10      1                   30          1000     0.9138036
##   0.10      1                   30          2000     0.9118195
##   0.10      1                   40           500     0.9057875
##   0.10      1                   40          1000     0.9057875
##   0.10      1                   40          2000     0.9018034
##   0.10      2                   20           500     0.9178356
##   0.10      2                   20          1000     0.9137875
##   0.10      2                   20          2000     0.9138034
##   0.10      2                   30           500     0.9138674
##   0.10      2                   30          1000     0.9037870
##   0.10      2                   30          2000     0.9038193
##   0.10      2                   40           500     0.9158356
##   0.10      2                   40          1000     0.9157555
##   0.10      2                   40          2000     0.9077873
##   0.10      3                   20           500     0.8957384
##   0.10      3                   20          1000     0.8977386
##   0.10      3                   20          2000     0.8977867
##   0.10      3                   30           500     0.8998349
##   0.10      3                   30          1000     0.8957867
##   0.10      3                   30          2000     0.8997867
##   0.10      3                   40           500     0.9057870
##   0.10      3                   40          1000     0.9037870
##   0.10      3                   40          2000     0.9018193
##   Kappa
##   0.2477796
##   0.2896310
##   0.3453436
##   0.2477796
##   0.2749786
##   0.3922098
##   0.2534232
##   0.2808042
##   0.3469924
##   0.3262847
##   0.3811835
##   0.3985153
##   0.3655727
##   0.4067450
##   0.3979521
##   0.3081609
##   0.3744329
##   0.4015308
##   0.3713709
##   0.3931424
##   0.4459542
##   0.3447254
##   0.4251822
##   0.4329775
##   0.3680925
##   0.4248587
```

```
##   0.4395640
##   0.3453436
##   0.3473580
##   0.3469118
##   0.4145206
##   0.3933368
##   0.4226881
##   0.3425110
##   0.3629593
##   0.3629593
##   0.4579625
##   0.4737115
##   0.4779569
##   0.4325291
##   0.4586858
##   0.4174886
##   0.4238416
##   0.4333370
##   0.4570101
##   0.4348587
##   0.4252009
##   0.4109967
##   0.4189324
##   0.3899285
##   0.3880362
##   0.4258991
##   0.4021604
##   0.4093848
##   0.3473580
##   0.3469118
##   0.3991895
##   0.4149358
##   0.4226881
##   0.4249090
##   0.3629593
##   0.3629593
##   0.3499585
##   0.4730730
##   0.4707930
##   0.4904980
##   0.4438136
##   0.3706470
##   0.3963751
##   0.4605458
##   0.4762529
##   0.4470723
##   0.3896980
##   0.4199706
##   0.4235632
##   0.3921294
##   0.3945809
##   0.4263659
##   0.4153724
##   0.4260754
```

```
##   0.4185839
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were n.trees = 1000,
##  interaction.depth = 2, shrinkage = 0.05 and n.minobsinnode = 30.
```

XGBOOST

```r
set.seed(1234)
xgbmgrid<-expand.grid(
  nrounds=c(5,10,50),
  max_depth=6,
  eta=c(0.1,0.5,1,5),
  gamma=0,
  colsample_bytree=1,
  min_child_weight=c(5,10,20),
  subsample=1)

xgbm<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+fam
             data=students.df.s,
             method="xgbTree",
             trControl=control,
             tuneGrid=xgbmgrid,
             verbose=FALSE)

xgbm
```

```
## eXtreme Gradient Boosting
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
## Resampling results across tuning parameters:
##
##   eta  min_child_weight  nrounds  Accuracy   Kappa
##   0.1   5                 5       0.8958029  0.15136291
##   0.1   5                10       0.8977227  0.20931867
##   0.1   5                50       0.9037232  0.29443823
##   0.1  10                 5       0.9018193  0.00000000
##   0.1  10                10       0.9018193  0.00000000
##   0.1  10                50       0.9138039  0.25798314
##   0.1  20                 5       0.9018193  0.00000000
##   0.1  20                10       0.9018193  0.00000000
##   0.1  20                50       0.9018193  0.00000000
##   0.5   5                 5       0.9057232  0.32448228
##   0.5   5                10       0.9057552  0.34647055
##   0.5   5                50       0.9097555  0.37842008
##   0.5  10                 5       0.9078193  0.09400998
##   0.5  10                10       0.9238361  0.38611339
##   0.5  10                50       0.9177878  0.37578408
##   0.5  20                 5       0.9018193  0.00000000
```

```
##   0.5  20               10       0.9018193  0.00000000
##   0.5  20               50       0.9018193  0.00000000
##   1.0   5                5       0.9117552  0.45423984
##   1.0   5               10       0.9077230  0.38769936
##   1.0   5               50       0.9077552  0.38817868
##   1.0  10                5       0.9097394  0.30435793
##   1.0  10               10       0.9077555  0.29834134
##   1.0  10               50       0.9077555  0.29834134
##   1.0  20                5       0.9018193  0.00000000
##   1.0  20               10       0.9018193  0.00000000
##   1.0  20               50       0.9018193  0.00000000
##   5.0   5                5       0.9018193  0.00000000
##   5.0   5               10       0.7002063  0.00000000
##   5.0   5               50       0.9018193  0.00000000
##   5.0  10                5       0.9018193  0.00000000
##   5.0  10               10       0.9018193  0.00000000
##   5.0  10               50       0.9018193  0.00000000
##   5.0  20                5       0.9018193  0.00000000
##   5.0  20               10       0.9018193  0.00000000
##   5.0  20               50       0.9018193  0.00000000
##
## Tuning parameter 'max_depth' was held constant at a value of 6
##
## Tuning parameter 'colsample_bytree' was held constant at a value of
##  1
## Tuning parameter 'subsample' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were nrounds = 10, max_depth = 6,
##  eta = 0.5, gamma = 0, colsample_bytree = 1, min_child_weight = 10
##  and subsample = 1.
```

SUPPORT VECTOR MACHINE - LINEAR

```
set.seed(1234)

SVMgrid<-expand.grid(C=c(0.01,0.1,0.2,0.3,0.5))

SVMl<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+fam
            data=students.df.s,
            method="svmLinear",
            trControl=control,
            tuneGrid=SVMgrid,
            verbose=FALSE)

SVMl
```

```
## Support Vector Machines with Linear Kernel
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
```

```
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##   0.01  0.9018193  0.0000000
##   0.10  0.9037873  0.1743032
##   0.20  0.9117555  0.2905190
##   0.30  0.9117555  0.2905190
##   0.50  0.9097714  0.3518066
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 0.2.
```

SUPPORT VECTOR MACHINE - POLYNOMIAL

```r
set.seed(1234)
SVMgrid<-expand.grid(degree=c(1,2,3),
                     scale=c(5:7),
                     C=c(3:5))


SVMp<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+fam
           data=students.df.s,
           method="svmPoly",
           trControl=control,
           tuneGrid=SVMgrid,
           verbose=FALSE)
SVMp
```

```
## Support Vector Machines with Polynomial Kernel
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
## Resampling results across tuning parameters:
##
##   degree  scale  C  Accuracy   Kappa
##   1       5      3  0.9097714  0.3518066
##   1       5      4  0.9097714  0.3518066
##   1       5      5  0.9117875  0.3773113
##   1       6      3  0.9097714  0.3518066
##   1       6      4  0.9117875  0.3773113
##   1       6      5  0.9117875  0.3773113
##   1       7      3  0.9097714  0.3518066
##   1       7      4  0.9117875  0.3773113
##   1       7      5  0.9117875  0.3773113
##   2       5      3  0.8557527  0.2698057
##   2       5      4  0.8557527  0.2586413
##   2       5      5  0.8557527  0.2586413
##   2       6      3  0.8557847  0.2581598
##   2       6      4  0.8537686  0.2528319
##   2       6      5  0.8557686  0.2581489
```

```
##   2       7      3  0.8537686  0.2528319
##   2       7      4  0.8557686  0.2581489
##   2       7      5  0.8597847  0.2915594
##   3       5      3  0.8678336  0.3090095
##   3       5      4  0.8678336  0.3090095
##   3       5      5  0.8678336  0.3090095
##   3       6      3  0.8658495  0.3054816
##   3       6      4  0.8658495  0.3054816
##   3       6      5  0.8658495  0.3054816
##   3       7      3  0.8618333  0.2968563
##   3       7      4  0.8618333  0.2968563
##   3       7      5  0.8618333  0.2968563
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were degree = 1, scale = 6 and C = 4.
```

SUPPORT VECTOR MACHINE - RADIAL

```r
set.seed(1234)
SVMgrid<-expand.grid(sigma=c(0.01,0.05,0.1),
                     C=c(1:4))

SVMr<- train(factor(higher)~G1+age+studytime+G3+school.GP+famsup.yes+Mjob.health+schoolsup.yes+Walc+fam
            data=students.df.s,
            method="svmRadial",
            trControl=control,
            tuneGrid=SVMgrid,
            verbose=FALSE)
SVMr
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 499 samples
##  11 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold, repeated 1 times)
## Summary of sample sizes: 375, 375, 373, 374
## Resampling results across tuning parameters:
##
##   sigma  C  Accuracy   Kappa
##   0.01   1  0.9018193  0.0000000
##   0.01   2  0.9018193  0.0000000
##   0.01   3  0.9058195  0.1142377
##   0.01   4  0.9078036  0.1392645
##   0.05   1  0.9058354  0.1250663
##   0.05   2  0.9157878  0.3511362
##   0.05   3  0.9118195  0.3724447
##   0.05   4  0.9157878  0.4079531
##   0.10   1  0.9057873  0.2133005
##   0.10   2  0.9078193  0.3062066
##   0.10   3  0.9117875  0.3811351
##   0.10   4  0.9057552  0.3636929
##
```

```
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.05 and C = 2.
```

Realizamos una comparativa de la precisión todos los modelos anteriores

```r
nnettune <- bestTuneNnet(rednnet)
bagtune <- bag$results
rftune <- bestTuneRf(rf)
gbmtune <- bestTuneGbm(gbm)
xgbmtune <- bestTuneXgbm(xgbm)
svmltune <- bestTuneSVMl(SVMl)
svmptune <- bestTuneSVMp(SVMp)
svmrtune <- bestTuneSVMr(SVMr)

models = c(reg$method,
           nnettune$method,
           bagtune$method,
           rftune$method,
           gbmtune$method,
           xgbmtune$method,
           svmltune$method,
           svmptune$method,
           svmrtune$method)

accuracies = c(reg$results$Accuracy,
               nnettune$Accuracy,
               bagtune$Accuracy,
               rftune$Accuracy,
               gbmtune$Accuracy,
               xgbmtune$Accuracy,
               svmltune$Accuracy,
               svmptune$Accuracy,
               svmrtune$Accuracy)

comparation <- data.frame("Model" = models, "Accuracy" = accuracies)
comparation[order(comparation$Accuracy, decreasing = TRUE),]
```

```
##       Model  Accuracy
## 6   xgbTree 0.9238361
## 5       gbm 0.9178515
## 9 svmRadial 0.9157878
## 2    avNNet 0.9138226
## 8   svmPoly 0.9117875
## 7 svmLinear 0.9117555
## 4        rf 0.9117394
## 1       glm 0.9097716
## 3   bagging 0.9017069
```

PREPARACIÓN DE MODELOS PARA ENSAMBLADO

```
## ------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------
##
## Attaching package: 'plyr'
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
##
## Attaching package: 'reshape'
## The following object is masked from 'package:dplyr':
##
##     rename
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
##   size decay   bag  Accuracy     Kappa AccuracySD   KappaSD
## 1    3   0.1 FALSE 0.9110182 0.4071574 0.01723245 0.1705323
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
##   mtry  Accuracy     Kappa AccuracySD  KappaSD
## 1   11 0.9026053 0.2883494 0.02102609 0.178837
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
##   mtry  Accuracy     Kappa AccuracySD   KappaSD
## 1    3 0.9062085 0.2270512 0.01330715 0.1661336
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
##   n.minobsinnode shrinkage n.trees interaction.depth  Accuracy     Kappa
## 1             30      0.05    1000                 2 0.9050213 0.3962059
##   AccuracySD   KappaSD
## 1 0.01375029 0.1043372
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
```

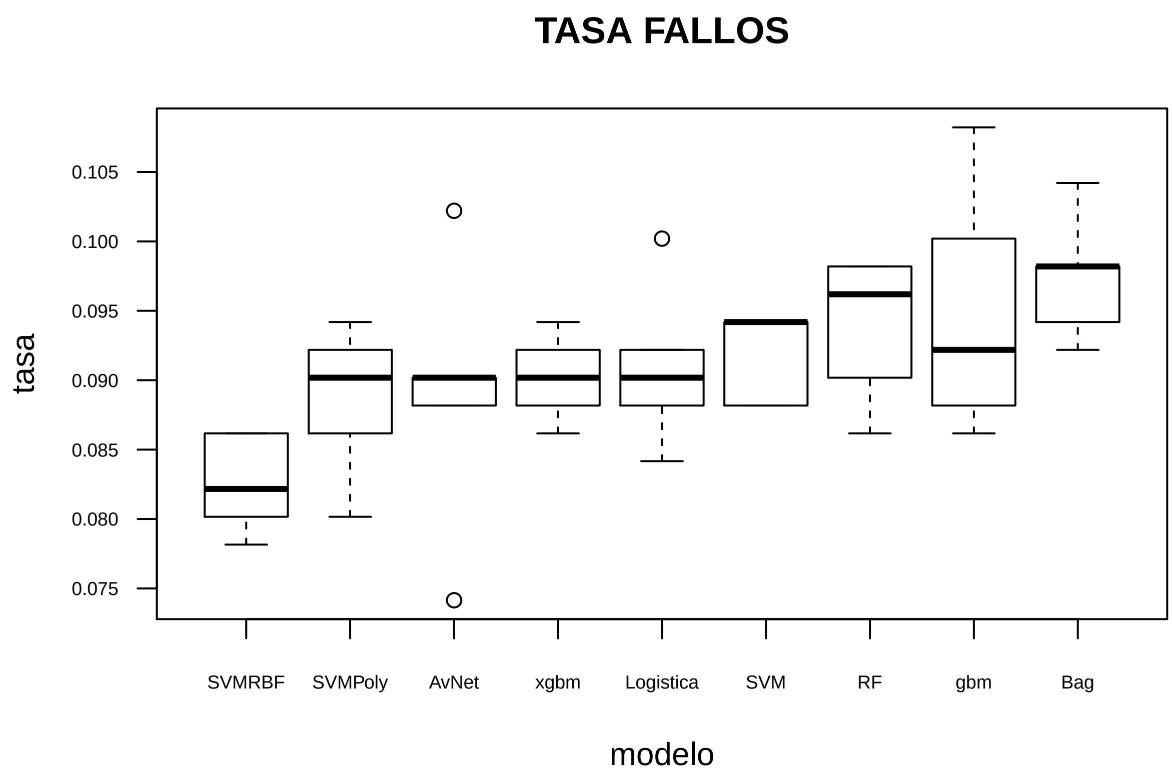
```
## Setting direction: controls < cases
```

```
## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
```

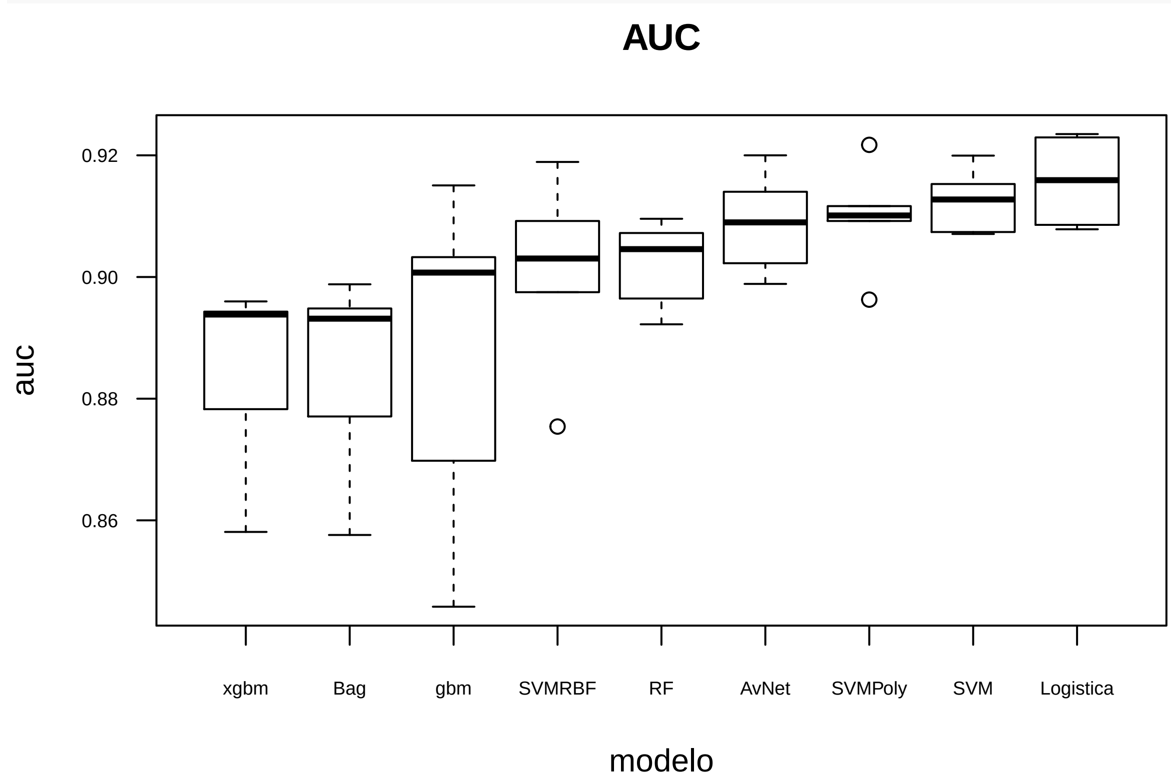
```
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:
##  objective
##   Only the last value for each of them will be used.

##   min_child_weight eta nrounds max_depth gamma colsample_bytree subsample
## 1               10 0.5      10         6     0                1         1
##    Accuracy     Kappa AccuracySD   KappaSD
## 1 0.9098118 0.2559003  0.0138007 0.1573665

## Setting levels: control = No, case = Yes
## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

##     C  Accuracy     Kappa AccuracySD   KappaSD
## 1 0.2 0.9082086 0.2493231  0.0149146 0.1830726

## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
##   C degree scale Accuracy     Kappa AccuracySD   KappaSD
## 1 4      1     6 0.911415 0.2365474 0.01347495 0.1735272
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
##   C sigma  Accuracy     Kappa AccuracySD   KappaSD
## 1 2  0.05 0.9174248 0.4037848 0.01362964 0.1728371
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```r
union$modelo <- with(union,
                       reorder(modelo,tasa, mean))
par(cex.axis=0.6, cex=1, las=1)
boxplot(data=union,tasa~modelo,main="TASA FALLOS")
```




```r
union$modelo <- with(union,
                     reorder(modelo,auc, mean))
par(cex.axis=0.6, cex=1, las=1)
boxplot(data=union,auc~modelo,main="AUC")
```



ENSAMBLADO DE MODELOS

```
unipredi<-cbind(predi1,predi2,predi3,predi4,predi5,predi6,predi7,predi8,predi9)
unipredi<- unipredi[, !duplicated(colnames(unipredi))]
```

```
unipredi$predi10<-(unipredi$logi+unipredi$svmPoly)/2
unipredi$predi11<-(unipredi$logi+unipredi$svmLinear)/2
unipredi$predi12<-(unipredi$logi+unipredi$avnnet)/2
unipredi$predi13<-(unipredi$svmLinear+unipredi$svmPoly)/2
unipredi$predi14<-(unipredi$avnnet+unipredi$svmRadial)/2


unipredi$predi20<-(unipredi$logi+unipredi$avnnet+unipredi$svmLinear)/3
unipredi$predi21<-(unipredi$logi+unipredi$avnnet+unipredi$svmPoly)/3
```

```
listado <- c("logi", "bagging", "avnnet", "rf",
"gbm", "xgbm", "svmLinear", "svmPoly", "svmRadial",
"predi10", "predi11", "predi12", "predi13", "predi14","predi20", "predi21")

listado
```

```
##  [1] "logi"      "bagging"   "avnnet"    "rf"        "gbm"
##  [6] "xgbm"      "svmLinear" "svmPoly"   "svmRadial" "predi10"
## [11] "predi11"   "predi12"   "predi13"   "predi14"   "predi20"
## [16] "predi21"
```

```
# Cambio a Yes, No, todas las predicciones

# Defino funcion tasafallos

tasafallos<-function(x,y) {
   confu<-confusionMatrix(x,y)
   tasa<-confu[[3]][1]
   return(tasa)
}

auc<-function(x,y) {
   curvaroc<-roc(response=x,predictor=y)
   auc<-curvaroc$auc
   return(auc)
  }

# Se obtiene el numero de repeticiones CV y se calculan las medias por repe en
# el data frame medias0

repeticiones<-nlevels(factor(unipredi$Rep))
unipredi$Rep<-as.factor(unipredi$Rep)
unipredi$Rep<-as.numeric(unipredi$Rep)


medias0<-data.frame(c())
for (prediccion in listado)
{
    unipredi$proba<-unipredi[,prediccion]
    unipredi[,prediccion]<-ifelse(unipredi[,prediccion]>0.5,"Yes","No")
    for (repe in 1:repeticiones)
```

```r
    {
        paso <- unipredi[(unipredi$Rep==repe),]
        pre<-factor(paso[,prediccion])
        archi<-paso[,c("proba","obs")]
        archi<-archi[order(archi$proba),]
        obs<-paso[,c("obs")]
        tasa=1-tasafallos(pre,obs)
        t<-as.data.frame(tasa)
        t$modelo<-prediccion
        auc<-auc(archi$obs,archi$proba)
        t$auc<-auc
        medias0<-rbind(medias0,t)
    }
}
```

```
## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases
```

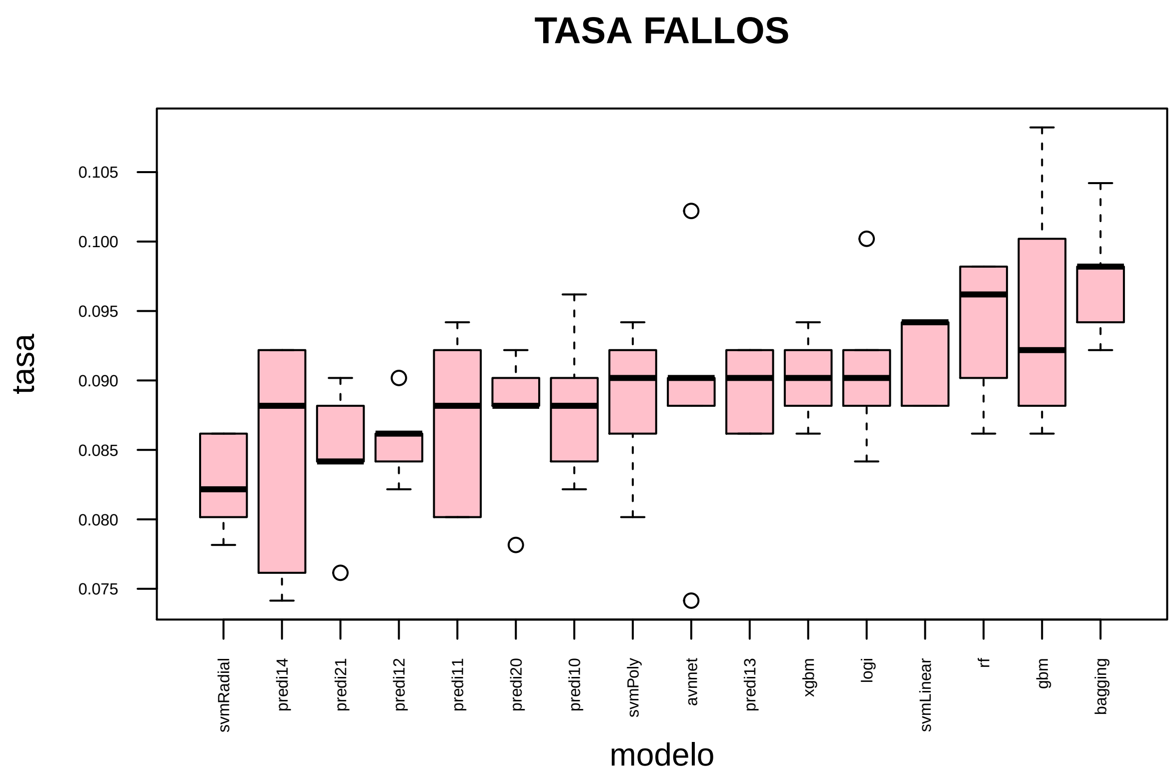
```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

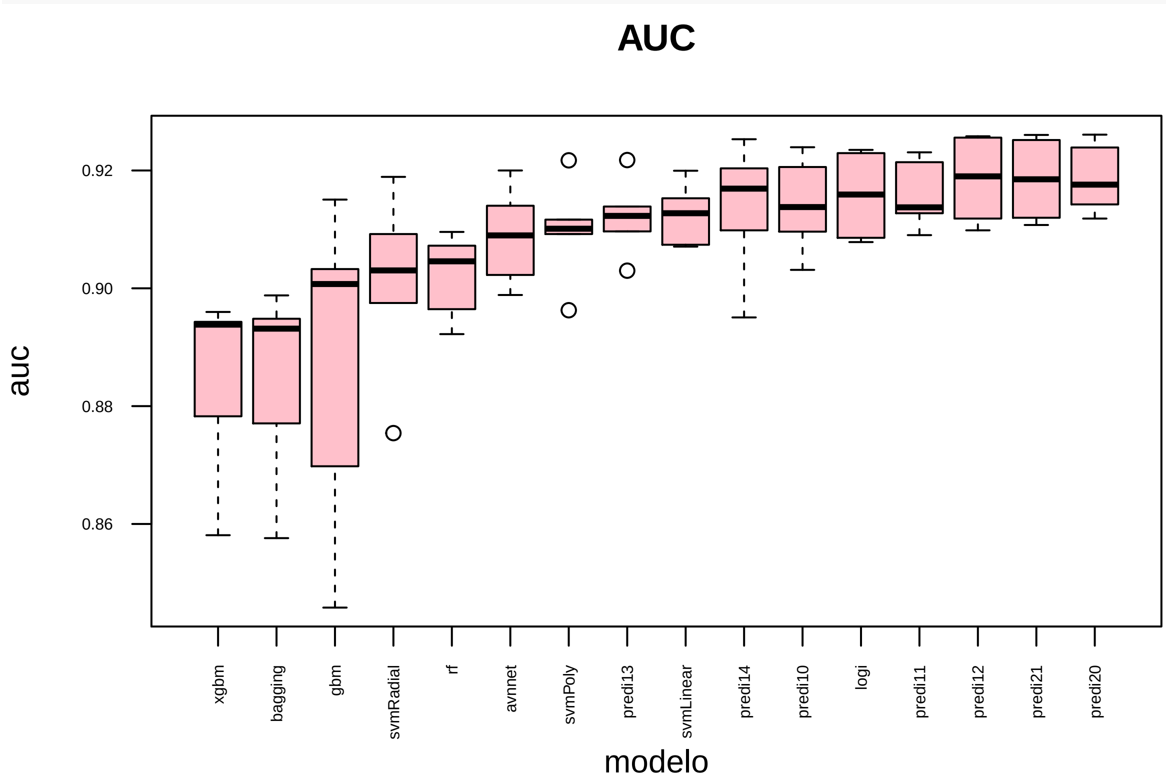
```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

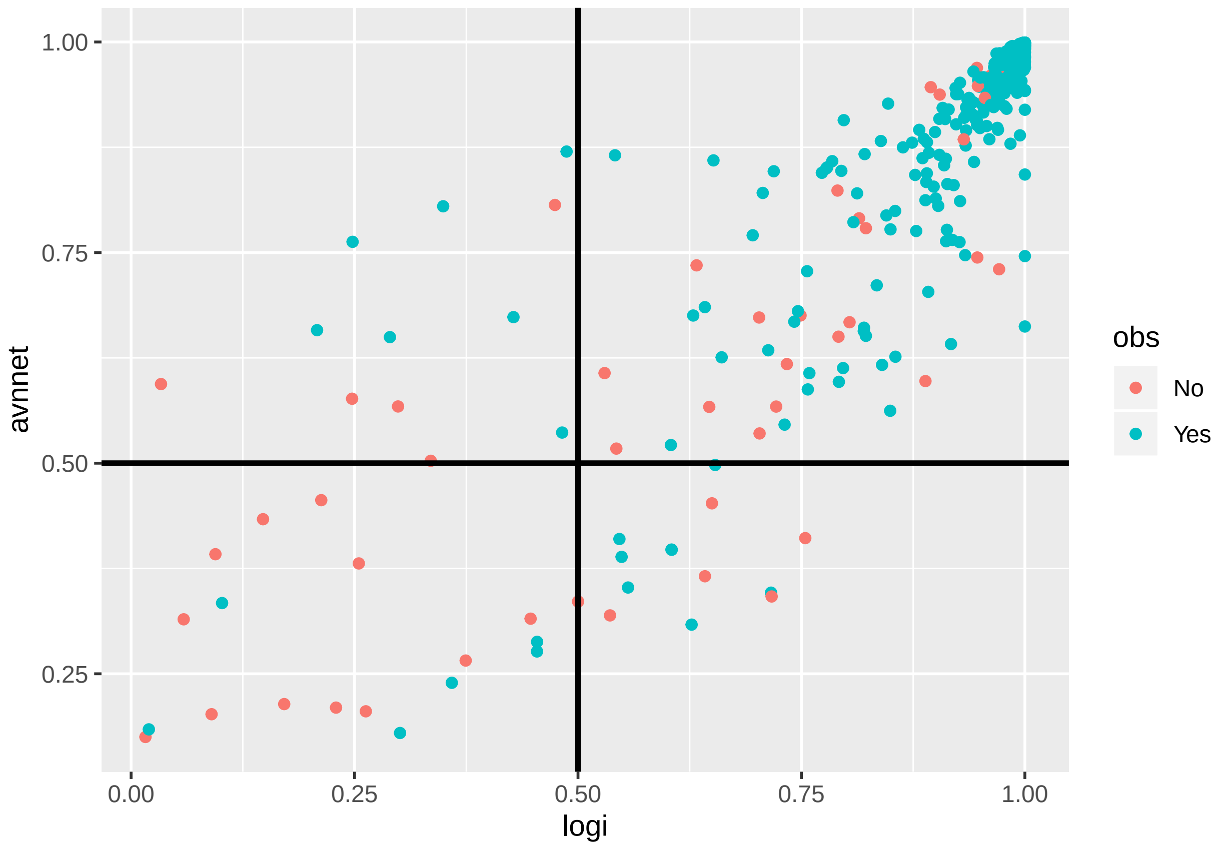
```r
# Finalmente boxplot
medias0$modelo <- with(medias0,
                       reorder(modelo,tasa, mean))
par(cex.axis=0.5,las=2)
boxplot(data=medias0,tasa~modelo,col="pink",main="TASA FALLOS")
```



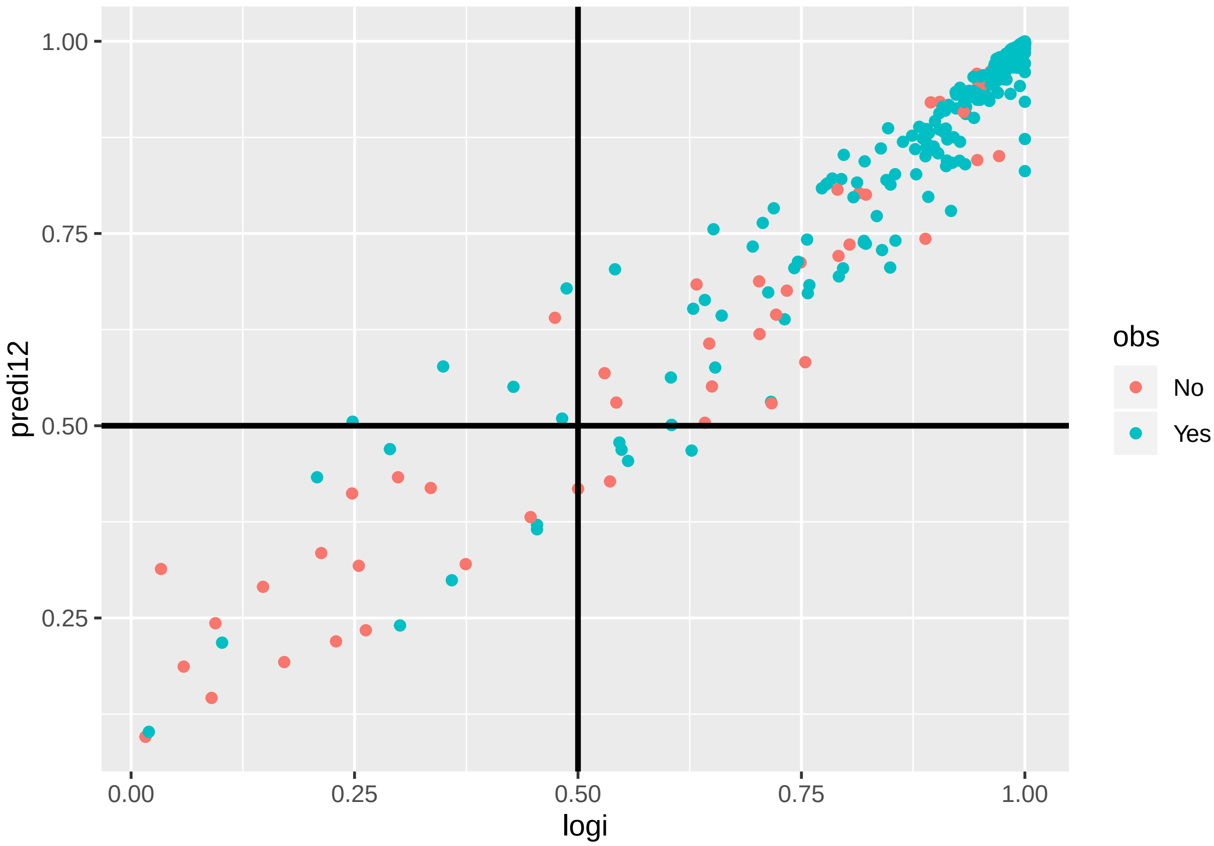

```r
# Para AUC se utiliza la variable auc del archivo medias0
medias0$modelo <- with(medias0,
                       reorder(modelo,auc, mean))
par(cex.axis=0.5,las=2)
boxplot(data=medias0,auc~modelo,col="pink",main="AUC")
```



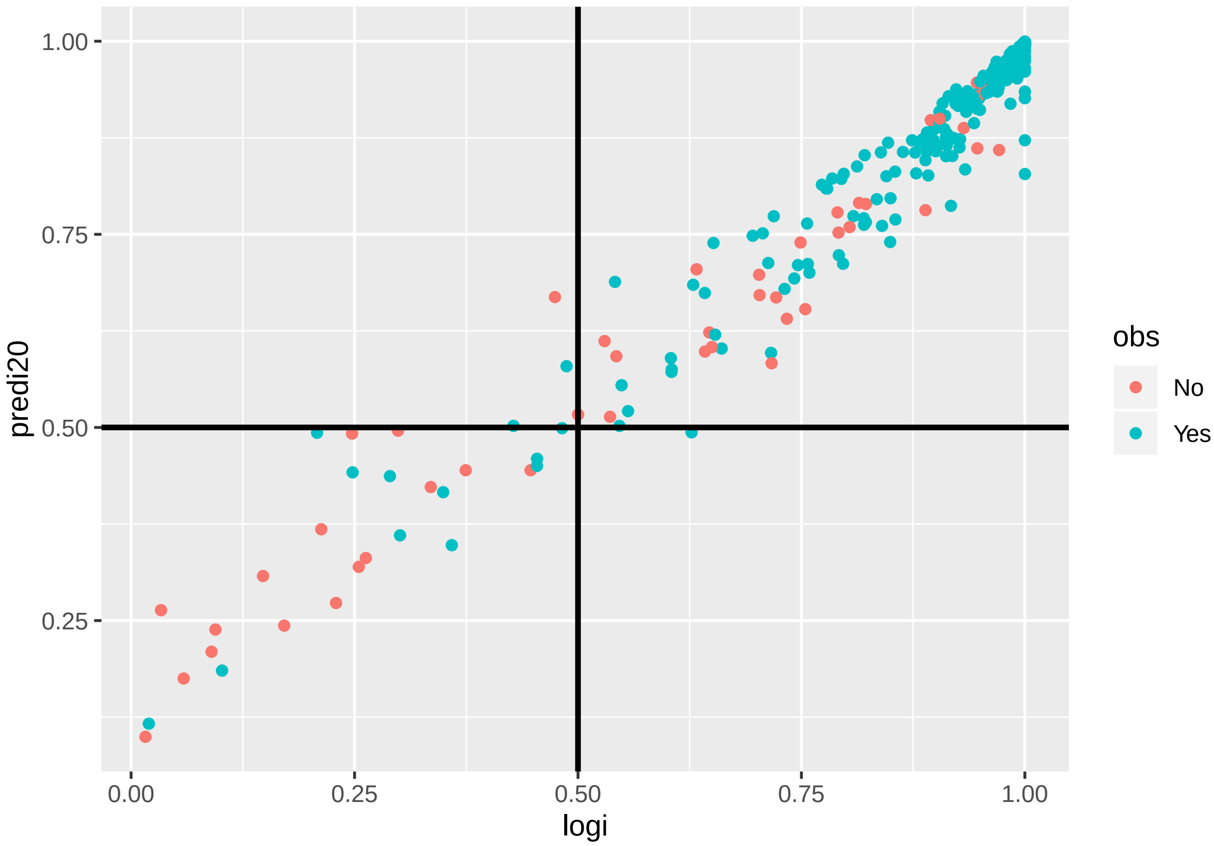
```r
unipredi<-cbind(predi1,predi2,predi3,predi4,predi5,predi6,predi7,predi8,predi9)
unipredi<- unipredi[, !duplicated(colnames(unipredi))]

unipredi$predi12<-(unipredi$logi+unipredi$avnnet)/2

unipredi$predi20<-(unipredi$logi+unipredi$avnnet+unipredi$svmLinear)/3
unipredi$predi21<-(unipredi$logi+unipredi$avnnet+unipredi$svmPoly)/3

unigraf<-unipredi[unipredi$Rep=="Rep1",]

qplot(logi,avnnet,data=unigraf,colour=obs)+
 geom_hline(yintercept=0.5, color="black", size=1)+
 geom_vline(xintercept=0.5, color="black", size=1)
```



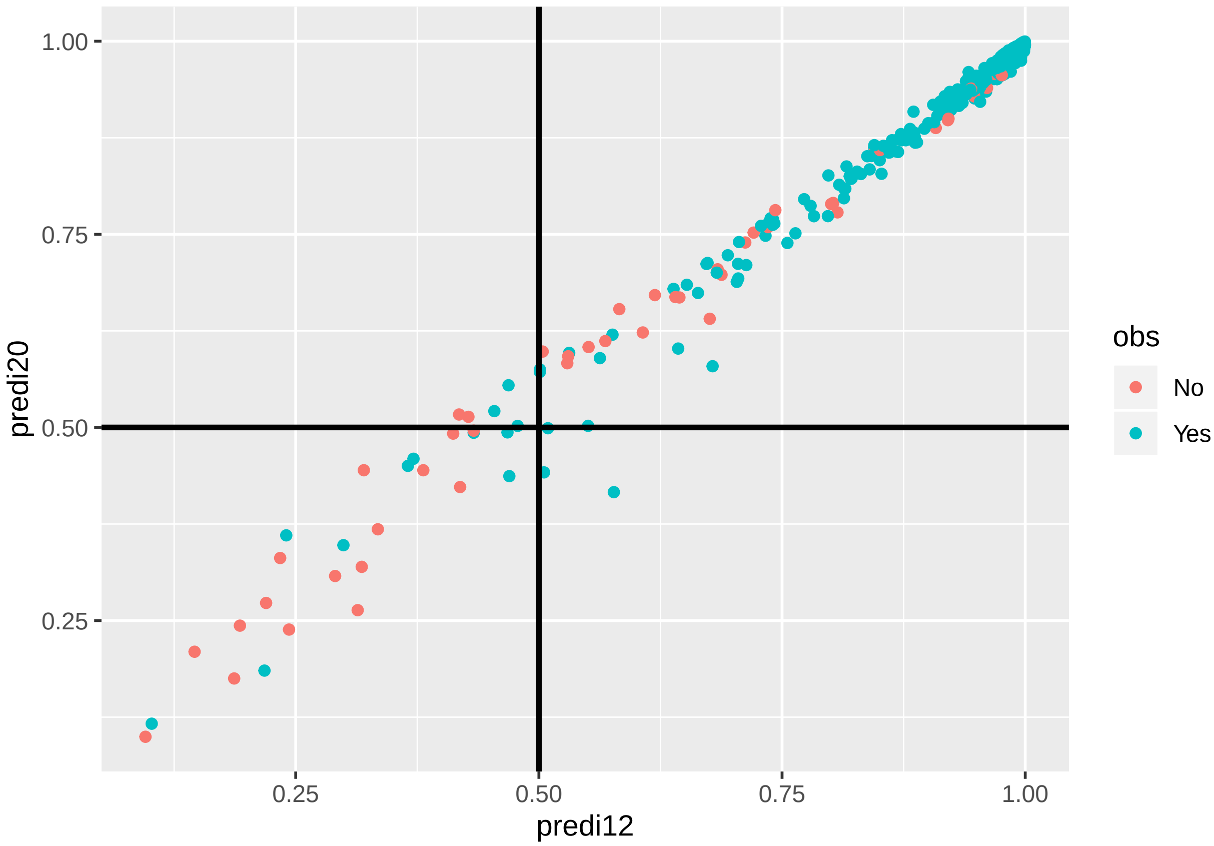

```r
qplot(logi,predi12,data=unigraf,colour=obs)+
 geom_hline(yintercept=0.5, color="black", size=1)+
 geom_vline(xintercept=0.5, color="black", size=1)
```



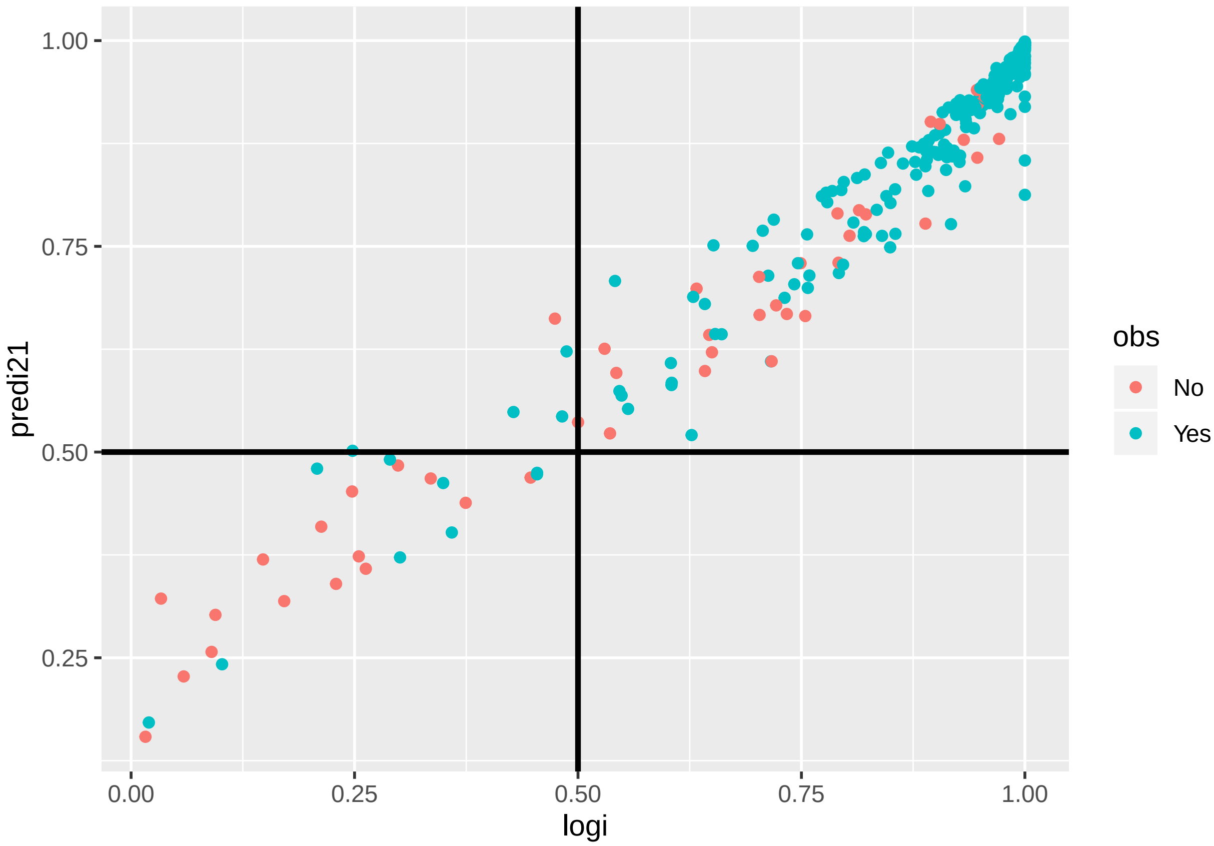

```
qplot(logi,predi20,data=unigraf,colour=obs)+
 geom_hline(yintercept=0.5, color="black", size=1)+
 geom_vline(xintercept=0.5, color="black", size=1)
```




```r
qplot(predi12,predi20,data=unigraf,colour=obs)+
 geom_hline(yintercept=0.5, color="black", size=1)+
 geom_vline(xintercept=0.5, color="black", size=1)
```




```
qplot(logi,predi21,data=unigraf,colour=obs)+
 geom_hline(yintercept=0.5, color="black", size=1)+
 geom_vline(xintercept=0.5, color="black", size=1)
```



CONCLUSIONES

El trabajo realizado